AMENDMENTS TO THE CLAIMS

This listing of claims replaces all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (Currently Amended) In a computer system including system memory, a method for providing information related to the termination of a process, the method comprising:

an act of loading a termination function into system memory, the termination function having termination instructions that, when executed, cause a calling process to terminate without providing information related to a termination event that caused the calling process to terminate;

an act of altering the code of the termination function to redirect the functionality of the termination function to a memory resident invalid instruction ~~detour function~~ such that a termination event ~~when the termination function is called instructions of the detour function are executed, the detour function having an~~ causes the invalid instruction ~~that, when~~ to be executed, execution of the invalid instruction causinges an exception that ~~can~~ provides termination information related to ~~a~~ the termination event ~~that would otherwise cause a calling process to terminate~~;

an act of a memory resident process detecting a termination event;

an act of the memory resident process calling the termination function; and

an act of executing the invalid instruction to provide termination information related to the detected termination event, in response to the termination function being called.

2. (Original) The method as recited in claim 1, wherein the act of loading a termination function into system memory comprises an act loading an application process into system memory.

3. (Original) The method as recited in claim 1, wherein the act of loading a termination function into system memory comprises an act loading termination function wherein the termination function is a terminate function, an abort function, an exit function, an _exit

function, a _cexit function, a _c_exit function, an _amsg_exit function, or an ExitProcess function.

4. (Original) The method as recited in claim 1, wherein the act of loading a termination function into system memory comprises an act loading a termination function in response to user input.

5. (Currently Amended) The method as recited in claim 1, wherein the act of altering the code of the termination function to redirecting the functionality of the termination function to a memory resident invalid instruction ~~detour function~~ comprises an act of activating in memory redirection.

6. (Currently Amended) The method as recited in claim 1, wherein the act of altering the code of the termination function to redirecting the functionality of the termination function to an invalid instruction~~memory resident detour function~~ comprises an act of redirecting the termination function to an ~~detour function~~ invalid instruction that can cause an exception having an increased likelihood of being propagated to an operating system code layer.

7. (Currently Amended) The method as recited in claim 6, wherein the act of altering the code of the termination function to redirecting the termination function to an invalid instruction ~~detour function~~ that can cause an exception having an increased likelihood of being propagated to an operating system code layer comprises an act of redirecting the termination function to ~~a detour function that includes~~ an instruction that attempts to write data to memory address zero.

8. (Currently Amended) The method as recited in claim 6, wherein the act of altering the code of the termination function to redirecting the termination function to an invalid instruction ~~detour function~~ that can cause an exception having an increased likelihood of being propagated to an operating system code layer comprises an act of redirecting the termination function to ~~a detour function including~~ an instruction that causes an access violation.

9. (Original) The method as recited in claim 1, wherein the act of a memory resident process detecting a termination event comprises an act of memory resident process detecting a C++ exception that was generated by the memory resident process, the C++ exception being an exception that cannot be appropriately handled by a debugger.

10. (Original) The method as recited in claim 1, wherein an act of executing the invalid instruction to provide termination information related to the detected termination comprises an act of generating an exception that has an increased chance of being propagated to an operating system code layer.

11. (Original) The method as recited in claim 1, wherein an act of executing the invalid instruction to provide termination information related to the detected termination event comprises executing an instruction that attempts to write data to memory address zero.

12. (Original) The method as recited in claim 1, wherein an act of executing the invalid instruction to provide termination information related to the detected termination event comprises an act of providing termination information wherein the termination is one or more of register values, a memory dump, and an event log.

13. (Original) The method as recited in claim 1, wherein an act of executing the invalid instruction to provide termination information comprises an act of an exception catcher catching an unhandled exception generated by the invalid instruction.

14. (Original) The method as recited in claim 1, wherein an act of executing the invalid instruction to provide termination information comprises an act of invoking a debug process that gathers termination information.

15. (Original) The method as recited in claim 1, further comprising
an act of activating in memory redirection in a system registry.

16. (Currently Amended) In a computer system including system memory, a method for providing information related to the termination of a process, the method comprising:

a step for configuring a termination function to execute an invalid instruction that provides termination information associated with a termination event, configuring the termination function including altering code such that a termination event causes the invalid instruction to be executed;

an act of a memory resident process detecting a termination event;

an act of the memory resident process calling the termination function; and

an act of executing the invalid instruction to provide termination information related to the detected termination event, in response to the termination function being called.

17. (Original) The method as recited in claim 16, wherein the step for configuring a termination function to execute an invalid instruction that provides termination information comprises a corresponding act of altering source code such that a termination event causes the invalid instruction to be executed.

18. (Original) The method as recited in claim 16, wherein the step for configuring a termination function to execute an invalid instruction that provides termination information comprises a corresponding act of altering binary code such that a termination event causes the invalid instruction to be executed.

19. (Original) The method as recited in claim 16, wherein the step for configuring a termination function to execute an exception instruction that provides termination information comprises a corresponding act of redirecting a library such that a termination event causes the exception instruction to be executed.

20. (Original) The method as recited in claim 19, wherein the corresponding act of redirecting a library such that a termination event causes the invalid instruction to be executed comprises an act of redirecting a dynamic link library.

21. (Currently Amended) A computer program product for use in a computer system including system memory, the computer program product for implementing a method for providing information related to the termination of a process, the computer program product comprising one or more computer-readable media having stored thereon computer executable instructions that, when executed by a processor, cause the computer system to perform the following:

load a termination function into system memory, the termination function having termination instructions that, when executed, cause a calling process to terminate without providing information related to a termination event that caused the calling process to terminate;

alter the code of the termination function to redirect the functionality of the termination function to a memory resident invalid instruction ~~detour function~~ such that a termination event ~~when the termination function is called instructions of the detour function are executed, the detour function having an~~ causes the invalid instruction ~~that, when~~ to be executed, execution of the invalid instruction causinges an exception that ~~can~~ provides termination information related to ~~a~~ the termination event ~~that caused a calling process to terminate~~;

detect a termination event;

call the termination function; and

execute the invalid instruction to provide termination information related to the detected termination event, in response to the termination function being called.

22. (Original) The computer program product as recited in claim 21, wherein computer-executable instructions that, when executed, cause the computer system to load a termination function into system memory comprise computer-executable instructions that, when executed, cause the computer system to load a termination function wherein the termination function is a terminate function, an abort function, an exit function, an _exit function, a _cexit function, a _c_exit function, an _amsg_exit function, or an ExitProcess function.

23. (Currently Amended) The computer program product as recited in claim 21, wherein computer-executable instructions that, when executed, cause the computer system to alter the code of the termination function to redirect the functionality of the termination function to a memory resident invalid instruction ~~detour function~~ comprise computer-executable instructions that, when executed, cause the computer system to ~~execute~~ insert a detour call into the termination function, the detour call calling the invalid instruction ~~detour function~~.

24. (Currently Amended) The computer program product as recited in claim 21, wherein computer-executable instructions that, when executed, cause the computer system to alter the code of the termination function to redirect the functionality of the termination function to a memory resident invalid instruction ~~detour function~~ comprise computer-executable instructions that, when executed, cause the computer system alter the code of the termination function to redirect the functionality of the termination function to a memory resident ~~detour function that includes an~~ nstruction that attempts to write data to memory address zero.

25. (Currently Amended) The computer program product as recited in claim 21, wherein computer-executable instructions that, when executed, cause the computer system to alter the code of the termination function to redirect the functionality of the termination function to an invalid instruction ~~memory resident detour function~~ comprise computer-executable instructions that, when executed, cause the computer system alter the code of the termination function to redirect the functionality of the termination function to a memory resident ~~detour function including an~~ instruction that causes an access violation.

26. (Original) The computer program product as recited in claim 21, wherein computer-executable instructions that, when executed, cause the computer system to execute the invalid instruction to provide termination information related to the detected termination event comprise computer-executable instructions, that when, executed cause the computer system to execute an instruction that causes an access violation.

27. (Original) The computer program product as recited in claim 21, wherein computer-executable instructions that, when executed, cause the computer system to execute the invalid

instruction to provide termination information related to the detected termination event comprise computer-executable instructions that, when executed, cause an exception having an increased chance of being propagated to an operating system code layer.

28. (Original) The computer program product as recited in claim 21, wherein computer-executable instructions that, when executed, cause the computer system to execute the invalid instruction to provide termination information related to the detected termination event comprise computer-executable instructions, that when, executed cause the computer system to provide termination information wherein the termination information is one or more of register values, a memory dump, and an event log.

29. (Original) The computer program product as recited in claim 21, wherein computer-executable instructions that, when executed, cause the computer system to execute the invalid instruction to provide termination information related to the detected termination event comprise computer-executable instructions that, when executed, cause a debug process to gather termination information.

30. (Original) The computer program product as recited in claim 21, wherein the one or more computer-readable media comprise physical storage media.